A Lean Approach to Improving SE Visibility in Large Operational Systems Evolution

Richard Turner
Stevens Institute of Technology
rturner@stevens.edu

Jo Ann Lane, Dan Ingold
University of Southern California
<jolane, dingold>@usc.edu

Ray Madachy
Naval Postgraduate School
rjmadach@nps.edu

Abstract. It is often difficult to understand the status of capability development in large operational systems. Schedules are rarely stable. This is due to factors such as: the size and complexity of capabilities; unexpected changes in priorities; depth of supplier chains; variety and availability of special engineering resources; contract structure; and the inherently complex nature of such operations. Lean approaches use the concepts of *work in progress (WIP)* and *capacity* to maximize flow through a process. Under certain circumstances, these concepts can be applied to systems engineering and development processes. This paper describes an example implementation of the concept in a large health care system of systems. To enhance both visibility and flow, the approach utilizes visualization techniques, pull-scheduling processes, and a services approach to systems engineering.

Introduction

Software development projects have successfully realized the benefits of agile and lean approaches (Anderson 2010, Reinertsen 2010, Poppendieck 2007, Larman and Vodde 2009, Boehm and Turner 2003). Improving the effectiveness of systems engineering in evolving operational systems seemed a good candidate for similar techniques. (Turner and Wade 2011) In 2011, a project was started to determine if pull scheduling might be more effective than complex integrated master schedules to manage (schedule and monitor) the systems engineering activities in such instances. An initial generalization of pull concepts using a standard kanban approach was developed. During the development, it was determined that to be effective, the definition of work items must closely resembled that of a service. The result was the description of a Kanban-based Scheduling System (KSS) (Turner, Lane, et al. 2012).

The second phase of this research is describing an implementation of the KSS concept to the development and maintenance of an information support system of systems (Dahmann et al. 2011, OUSD 2009) for a hypothetical large multi-facility hospital system. The example implementation uses a network of integrated KSSs (KSS Network) that are intended to:

- Make visible work in progress
- Establish and track organizational capacities at all levels
- Limit WIP to improve flow (identify resource issues, cause of blocked work)
- Coordinate multiple levels of systems engineering activity
- Communicate progress with respect to senior management goals
- Support analysis and decision making at every level of management
- Make visible current progress toward development and deployment of capabilities
- Establish a basis for continuous improvement in a rapidly changing environment

The KSS Network shows the relationships between the software development tasks and the systems engineering tasks. It also clearly captures the relationships between both the

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **24 JUN 2013**	2. REPORT TYPE	3. DATES COVERED **00-00-2013 to 00-00-2013**

4. TITLE AND SUBTITLE **A Lean Approach to Improving SE Visibility in Large Operational Systems Evolution**	5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Stevens Institute of Technology,Castle Point,Hoboken,NH,07030**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
23rd Annual INCOSE International Symposium, 24-27 June 2013, Philadelphia, PA.

14. ABSTRACT
It is often difficult to understand the status of capability development in large operational systems. Schedules are rarely stable. This is due to factors such as: the size and complexity of capabilities; unexpected changes in priorities; depth of supplier chains; variety and availability of special engineering resources; contract structure; and the inherently complex nature of such operations. Lean approaches use the concepts of work in progress (WIP) and capacity to maximize flow through a process. Under certain circumstances, these concepts can be applied to systems engineering and development processes. This paper describes an example implementation of the concept in a large health care system of systems. To enhance both visibility and flow, the approach utilizes visualization techniques pull-scheduling processes, and a services approach to systems engineering.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**16**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

software and systems engineering tasks and the required capabilities. Because kanban concepts have been primarily used with single level value streams, we wanted to understand the information needed for decision making, including scheduling and flow monitoring/control, at each level of SE activity or utilization. This would allow us to construct a KSS that would support visualization of WIP and status for each specific level. It would also provide insight into the information flow required.

To accomplish this, we turned to the Goal Question Metric approach (Basili and Seaman 2010). We defined the goals and the questions to ask to determine if the goals were being met. Given our research is to investigate KSSs, we decided to determine the information that each KSS would provide, referring to the approach as goal-question-kanban (G-Q-K). We will continue evolving the G-Q-K within the team and through the working group.

A KSS is a means of visually controlling workflow. It consists of a set of activities, where each activity has its own ready queue, a set of resources to add value to work units that flow through it, and a done queue. Visual representation provides immediate understanding of the state of flow through the set of activities. This transparency makes resource issues and process anomalies (both common and special cause) easily visible, enabling the team to recognize and react immediately to resolve issues locally. Because the team and management interact with the visualization and collectively solve problems, this aspect is important in achieving continuous improvement (kaizen). Control of the KSS is generally maintained through *WIP limits, small batch siz*e, and *Classes-of-Service* (COS) definitions that prioritize work with respect to risk. WIP is partially completed work, and in knowledge work can be roughly associated to the number of work items that have been started and not delivered. *WIP Limits* specifically cap the amount of work assigned to a set of resources. This lowers the context-switching overhead for individuals or teams performing simultaneous work items; accelerates delivery of value by completing work in progress before starting new work; and provides for reasonable and sustainable resource work loads. Classes of service (CoSs) provide a variety of handling options for different types of work items and affect the next task selection value of specific items of work for KSSs.

The idea of applying SE as a service within an on-demand scheduling system is an attempt to merge the SE flow and the software development project flow rather than simply lay SE functions on top of project activities without concern for the rapid-response constraints. In general, systems engineering is involved in three kinds of activities in rapid response environments: lifecycle, continuous, and requested. Lifecycle activities include front-end work like creating operational concepts, defining architectures, capability and requirement decomposition and allocation, verification, validation integration, and deployment activities. Continuous SE activities are ongoing, system–level activities (e.g. architecture analysis, performance analysis, configuration and risk management, and incremental verification, validation and integration). These require not only substantial time, but also the maintenance and evolution of long-term, persistent artifacts that support development across multiple projects. Requested activities are generally specific to either individual projects or capability engineering (e.g. issue triage, trade studies, impact assessments, needs analyses, cost analyses, interface support, and specialty engineering support), and draw on the persistent SE artifacts and knowledge.

By viewing persistent artifacts as key components of services provided to various projects, SE can be opportunistic in applying its cross-project view and understanding of the larger environment to specific projects individually or in groups. It can also broker information between individual projects where there may be contractual or access barriers. When a system-wide issue or external change occurs, SE can negotiate or unilaterally add or modify work items within affected projects to ensure that the broader issue is handled in an effective and compatible way. The quality of a service may be pre-specified, specified as a

parameter of the service request, or negotiated as a function of typical value and time available to provide the service. SE services may be thought of as a single activity, although some activities, particularly those up front, are likely to be complex enough to have their own set of value adding activities and specialized resources.

To support timeliness, SE performs its services in parallel to those activities in the requesting project, prioritizing individual project work across the entire system, and then provides the results to the requestor as soon as available.

Hospital System Information Support Development

The health care SoS is a set of integrated medical information management systems. It consists of hardware, over two million lines of source code, numerous commercial-off-the shelf (COTS) software products, and communications networks that support the administration and delivery of health care in networked set of hospitals and clinics.

The Health Care Development Organization consists of three groups. The **systems engineering group** is responsible for make-versus-buy trade studies related to new capabilities or enhancements that might be provided by COTS products, conducting evaluations of candidate medical devices for integration into the health care SoS, system performance assessments of both deployed systems and system enhancements under development, networks for both the deployed systems and the development environments, hardware and network upgrade recommendations, security engineering, constituent system integration, and system and SoS-level acceptance testing. The **software development group** is responsible for software maintenance and enhancement for the custom Health Care constituent systems or products; database structures and embedded procedures; COTS product tailoring, glue code, integration, and upgrades; and licensed data upgrades such as the pharmacy approved formularies, as well as responding to software issues that are beyond the scope of the user help desk. The **user and site support group** is responsible for running the user help desk, site configuration management, and site installations and upgrades. In an organization such as this, there may be as many as 1,000 engineering professionals working on this system—about a third in software development.

The key custom software constituent systems within the health care SoS include user access management, patient management, pharmacy, laboratory, radiology, and patient telemetry. The constituent systems share a single database that maintains the information for all of the patients and personnel related to a given health care site. Some key constituents are supported by tailored COTS products and integrated into the health care system. In addition, there are interfaces to other health care systems maintained by the parent organization at various sites. The interfacing systems include custom legacy systems, COTS products, and electronic medical devices such as heart rate monitors and infusion pumps.

The health care system's primary goal is to support patient health care and to provide health care in a timely and safe manner that is coordinated across a variety of health care providers and specialists. Key overarching requirements are to ensure patient-safety and to protect patient information in accordance with government Health Insurance Portability and Accountability Act (HIPAA) and other privacy and security regulations. To meet these goals and regulations, the health care organization provides periodic software and system updates.

The current systems engineering and software engineering organizations are fully staffed with respect to development budget. When new needs or capabilities are identified, systems engineering analyzes the new needs/capabilities in terms of the given systems and decides how address them. Often multiple new needs/capabilities are analyzed together to facilitate the identification of common solutions that can support more than one need/capability as well as support performance upgrades and technology refresh. The results of the analysis activities are a set of requirements. The next step in the process is to allocate those requirements to one

or more products for implementation. Figure 1 provides an example that illustrates how multiple requirements are derived from one or more needs and then mapped to the enterprise products for implementation. It also shows how well-engineered capabilities can use common solutions and requirements can often map into more than one constituent system/product.

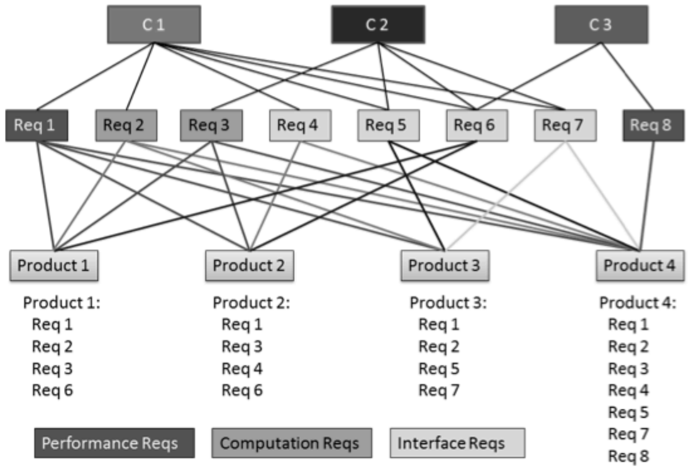

Figure 1. Capabilities to requirements to products

Once the requirements are allocated to the products, the product teams analyze them and convert them into features and stories for implementation. Systems engineering monitors the capability "pieces" to guide their system integration and testing activities. When all of the capability requirements are implemented in the affected products and deployed, the mission capability is considered "completed."

Several issues exist. There is no visibility at the capability level showing which user stories are related to which capabilities and which products are implementing pieces of the capability. The systems engineering resources are hampered by variable, multiple tasks, and rapidly changing priorities. Software tasks become blocked waiting for systems engineering tasks to complete. As a result, started tasks are difficult to complete in a timely manner.

The KSS Network Architecture

A new three-tiered KSS architecture is defined to improve the software development flow and to enhance senior management visibility into the development process.

The proposed organizational levels are:

- Executive/Stakeholder Management (ESM)
- Capability Engineering (CE)
- Product/Domain Engineering (PDE)

Figure 2 provides an overview of the Health Care System KSS-network.

Executive/Stakeholder Management Level. The ESM level determines which proposed capabilities (or capability enhancements) are going to be approved for development. As part of this process, the ESM level assesses the value of the capability against its expected cost and schedule to develop. This highest-level in the KSS network is concerned primarily with the current status of identified capabilities (or needs) as represented by the development state of each "not fully deployed" but "approved for development" capability – essentially WIP. At this level, the KSS is tracking capabilities and their priority. The insight it provides should inform decisions about overall organizational strategy, resource staffing, and development funding priorities.

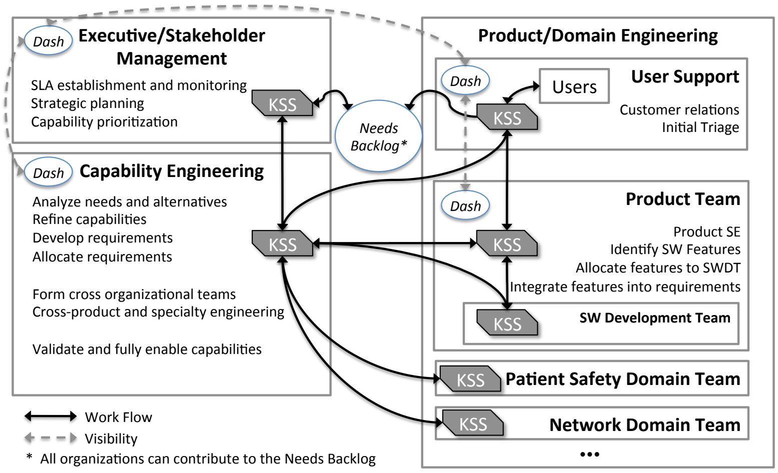

Figure 2. Overview of KSS Network

The *Executive/Stakeholder Management Level* determines which proposed capabilities (or capability enhancements) are going to be approved for development. As part of this process, ESM assesses the value of the capability against its expected cost and schedule to develop. This highest-level in the KSS network is concerned primarily with the current status of identified capabilities (or needs) as represented by the development state of each “not fully deployed” but “approved for development” capability – essentially WIP. At this level, the KSS is tracking capabilities and their priority. The insight it provides should inform decisions about overall organizational strategy, resource staffing, and development funding priorities.

Capability Engineering (CE) represents all capability-related SE activities, specialty SE support for product teams, including software system engineering tasks, where software is a key component in the requirements allocation. CE is responsible for creating capability descriptions that incorporate the needs identified and prioritized by the ESM level. CE must balance the various SE resources as they work with both internal activities and lead cross-organizational teams in CE-related activities. Decisions and scheduling of the SE resources must include front-end and ongoing architectural work as well as support to development, integration, verification and validation that interacts with product teams.

At the *Product/Domain Engineering* (PDE) level, there are separate KSSs for each product or domain team in the enterprise. The KSSs at this level are similar to those in use in many software development organizations today, with the added requirement to perform systems engineering within the product or domain scope. What is different for constituent systems/products that participate in one or more SoSs, is the need to provide information to higher level KSSs and dashboards all the way to the ESM level.

The US Team operates at the PDE level because it primarily interfaces with the product and domain teams. There are occasions, however, when it influences the needs backlogs, or when it uncovers an issue (e.g. patient safety or privacy) that requires engagement with ESM and CE to handle the solution. Each product or domain team is responsible for implementing capability-related requirements allocated to that product as well as responding to User Support problems that cannot be handled by the US team.

Each product team may have a unique team organization depending on whether it is internal or outsourced. If outsourced, contractual requirements, corporate size and corporate governance influence the KSS implementation. For example, if the outsourced organization operating the product team uses a matrix organization for SE, there may be a separate KSS defined for the SE resources that may cross product team boundaries. If the contractual SE resources are each dedicated to a specific product, then their tasks can be included in the

individual product team KSS or the software development team KSS.

Classes of service (CoSs) provide a variety of handling options for different types of work and affect the next work item selection value for KSSs. They may be aligned with Service Level Agreement (SLA) priorities. Most CoSs are intended to ensure priority rather than force immediate execution. There are CoSs that are disruptive–that is, they can suspend current work in progress. These are associated with critical or expedited work to allow swarming of all appropriate resources to ensure completion as soon as possible. However, disruptive CoSs are minimized because they counter the normal kanban philosophy of completing work rather than interrupting it. While most CoSs are shared across the entire KSS network, individual KSSs may define additional *KSS-Specific* CoSs to handle flow specific to their types of work. We have defined five general COSs that apply to all the work in the Health Care SoS KSS Network.

CoS	**Description**
Critical Expedite	Safety, security, or other emergency work items. Disruptive: requires necessary resources to stop current work and complete it.
Important	Very high priority work items such that this work takes priority over other work in the ready queue. Not Disruptive.
Date Certain	Work items that must be completed by a specific date or there will be significant consequences.
Standard	The normal CoS for the development organizations work.
Background	Work that must go on but is usually not time critical. It includes things like architectural enhancements, low-level technical debt, or research and environmental scanning

Table 1. General Classes of Service

KSS Description

Each KSS is based on the workflow, the G-Q-K definitions, and the special circumstances and needs of each organization of resources represented by the KSS. There are nearly as many ways to define a KSS as there are to define a system. We simply recommend processes and visualizations appropriate to our target organization. Going forward, we will try to identify patterns or anti-patterns that occur. Each KSS we describe includes a summary, process flow descriptions, and visualization tools. Table 2 is the summary template.

KSS Flow Process Descriptions provide insight into the information and activities within the KSS. There are several processes common to each KSS. Other processes may be identified for a specific KSS.

The process for *Accepting/Selecting Next Work Item* describes how work items are selected from the demand backlog. This is a highly personal and collaborative process closely related to the resource allocation process. It includes the cadence for selection, the limits of the demand queue and any limits on the backlog. This process addresses constraints on the capacity and on the demand that would impact the performance and balances the value of the work to both the demand source and to the performing KSS. The value of work may change due to adjustments related to the KSS environment. For example, some work sources may have inherent priority over other sources for political or other reasons. CoS may be interpreted differently where there is a higher instance of ongoing maintenance tasks so that critical maintenance is not pushed out to far by new capability or enhancement projects. Resources may also drive manipulating the value. If, for example, a significant number of resources are delegated to cross-organizational work or are absent for other reasons (e.g. military deployment or illness), there might be a lowering of the value of work that might require their expertise until such time as they return. Finally, there can be general rules. An example we incorporate here is a selection prejudice toward work affecting a requirement or

capability nearing completion. This encourages completing work and reducing WIP.

Table 2. KSS Template

KSS Name			
Demand:			
Work sources	*Organizations that can assign work items to the KSS*		
Resources:			
Dedicated	*Resources under control of this KSS*		
Shareable	*Resources available to share on teams with other orgs*		
Sourced	*Organizations (KSSs) to which work items can be assigned*		
Managed resource specialties	*Any specialists that are managed individually*		
Activities:			
Description	WIP Limit	Resource Type	Cohesion
		Internal, Sourced, or X-discipline team	*Interruptible or Must complete*
Flow and Visibility:			
Additional CoS handled	*CoS beyond the system-wide that are recognized by this KSS*		
Additional CoS introduced	*CoS defined for work this org assigns to other KSSs*		
Work Selection Value Adjustments			
Source-based	*CoS-based*	*Resource-based*	*Completion-based*
Goals	*From GQK analysis*		
Questions answered	*From GQK analysis*		
Data maintained/used	*From GQKanalysis*		
Information shared	*e.g. Avg. Lead time, Avg. blocked tasks. Avg. time blocked, Avg. resource WIP, Avg. Backlog length, Statistical limits for information types,*		

Allocating Resources and Team Development handles resource assignment and the formation of teams where required. Every KSS will have different processes for allocation of work. There may be specific assigned teams, first available resource, or self-selection. This process interacts with the selection process by considering the existing capacity to complete work in the demand queue.

Completion and Disbursement specifies any actions that are required when work items are placed in the KSS done queue. As an example, this could include work collecting and integrating sub-tasks derived from a work item and separately handled that must all be completed before the initial work item is considered "done."

KSS Review is the process for walking the visualization board or reviewing the dashboard. It sets the cadence for the review, describes the way status is reviewed and resource/blockage issues identified, and what decisions as to resource allocation, work item selection, and incremental process improvement are considered and made.

The two keys to the pull scheduling approach are the visibility into work in progress and the ability to resolve flow issues at the lowest levels. Visibility is dependent on the efficacy of the *Visualization Tools*. In the complex multi-level systems engineering and development environment, the visualization tools will almost certainly be interconnected electronically. There are two specific types of visualization tools used in the Health Care KSS Network.

Kanban boards are active management and control tools that provide a common operating picture for all resources and work items associated with a KSS. The boards are organized according to demand, activities, and status, and have work items as their predominant content. They are interactive and updated rapidly to act as both information radiators and operational tools where information is added, consulted, adjusted, and removed as the work flows through the systems. *Dashboards* are used where multiple KSSs need to be involved in resource management and decision-making. A dashboard shows information gathered by the KSSs, providing the information from the G-Q-K analysis. Dashboards are pure information radiators designed to quickly communicate status assessment and decision making data for the specific area or organizational component. Rarely interactive, they may feature data in context charts (such as graphs or percentages) or scrolling information.

The Healthcare KSS Network

Executive/Stakeholder Management KSS

The ESM KSS tracks development performance in achieving high value output as quickly as reasonable and in accordance with the established goals. Organizational performance is illustrated continuously on the kanban board and summarized by the Dashboard.

Table 3. ESM KSS Template

Executive/Stakeholder Management KSS			
Demand:			
Work sources	Needs backlog, Stakeholders, Critical Events, Strategic Plans		
Resources:			
Dedicated	IT Managers, CTO, …		
Shareable	None		
Sourced	CE		
Managed resource specialties	None		
Activities:			
Description	*WIP Limit*	*Resource Type*	*Cohesion*
Capability Analysis		Sourced (CE)	Interruptible
Capability Prioritization-CoS Assignment		Internal	Must complete
Capability Development Project		Sourced (CE)	Interruptible
Flow and Visibility:			
Additional CoS handled	None		
Additional CoS introduced	None		
Work Selection Value Adjustments			
Source-based	*CoS-based*	*Resource-based*	*Completion-based*
None	None	None	None
Goals	G1. Deploy capabilities according to value-based priorities and CoS. G2. Understand source/cause of blocked work flows G3. Strategic IT decisions based on current and projected WIPs and backlogs (examples might include investments in additional resources (hardware, tools, people) or decisions to drop lower priority capabilities). G4. Changing needs and priorities are integrated with existing strategy		
Questions answered	Q1. What capabilities are currently in progress? Q2: What capabilities are currently blocked? Q3: What capabilities are pending acceptance? Q4. Are the planned and actual values of each deployed capability tracking? Q5: Are the current WIP level for ESM activities correct? Q6. What is the average time to completion for "accepted" capabilities by CoS? Q7. What is the requirements volatility by capability? Q8. What KSSs show capacity not meeting demand? Q9: What KSSs indicate excess capacity?		
Data maintained/used	KSS1: Flow data on CE and Product Teams* KSS2: Average time to deploy capabilities for each CoS priority level KSS3: Relationships between capabilities and requirements KSS4: Status of requirement completion/deployment KSS5: Percentage of requirements completed/deployed for each in-process capability KSS6: Status of SE tasks supporting capability acceptance decisions *Includes CFD (throughput, WIP, Lead time), backlog level, resource utilization, blocked tasks, and similar data.		
Information shared	Capabilities under development, CFDs for each Capability, Network Value Tracking,		

Accepting/Selecting Next Work Item. Requests for system capabilities come from the users, systems engineering groups, and strategic initiatives. There is always a backlog of ideas needs, and wants. ESM must identify the highest priority capabilities. They must balance adding new capabilities with improving existing system capabilities and maintaining the infrastructure. They must also act on critical issues regarding patient safety, infrastructure failure, and regulatory changes. The outcome of this process is sending only the highest value and most critical work to the systems engineering group to analyze and develop.

Some work items initiated within the ESM level are special studies related to the

prioritization of capabilities and the possible combination of multiple needs into a more effective capability need. This work includes cost and schedule estimations, Ops Concept development, COTS evaluations, and other traditional front end SE activities.

Allocating Resources and Team Development. ESM must understand the overall capacity, work in progress, and resource distribution across CE and PDE teams in order to determine the highest priority capabilities and decide how to meet strategic needs and balance ongoing tasks. Starting too many capability developments can lead to less effective execution, while starting too few may jeopardize market share or stakeholder satisfaction. This organization must work closely with the CE organization and the User Support Team to map the landscape reflected in the needs backlog.

Completion and Disbursement. While the decision to deploy is often a systems engineering or PDE Team decision, the declaration of a capability being "finished" (i.e. fully implemented and deployed) is usually reserved for the ESM.

KSS review at this level examines the work in progress, demand, capacity, and performance to ensure it is focused on achieving capabilities and handling critical events. Resource management, including budgeting, requires an understanding of how development resources are being utilized throughout the system, what is in the backlog of desired capabilities, and areas where there is excess capacity or capacity is insufficient for the projected demand. Budgeting is a factor in determining how much demand is realistic regardless of capacity. Strategic changes to resource mix across the system of systems may be needed to improve flow and are made through hiring, contracting, or moving resources.

Backlog (Demand)	Capability Analysis		Capability Development		Done

Figure 3. ESM Kanban Template

ESM Dashboard Capabilities in Progress	CoS	Value	Total # of Requirements	# Requirements Completed Last Month	This Month	% Value completed	# Requirements in Progress	% Value in Progress	% Requirements with work items blocked	Expected Completion
Capability 1										
Capability 2										
Capability 3										
Capability 4 (CRITICAL)										
Capability 5										

ESM Backlog	Items in backlog CoS	Value
Capability 6		
Capability 7		
Capability 8		
Capability 9		
Capability 10		
Capability 11		
Capability 12		
Capability 13		

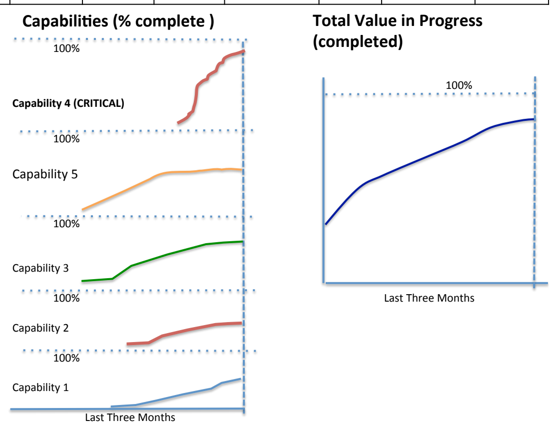

Figure 4. Notional Executive/Stakeholder Management dashboard

Capability Engineering (CE) KSS

The CE KSS represents multiple levels of activity and as the complexity grows may choose to break into multiple KSSs. However, the initial concept is a single KSS that handles a variety of different activities. First and foremost, the CE must respond to the ESM requests

for both analysis and SE support to ESM decision activities and for the development of capabilities that are the highest priority to the SoS. On a secondary note, the CE provides SoS analysis support to the various PDE Teams and manages the limited number of SoS specialty engineering resources. Given the goals associated with this level, both the kanban board and the dashboard will be somewhat "busy" in terms of information.

Table 4. CE KSS Template

Capability Engineering KSS	
Demand:	
Work sources	ESM, PDT, Internal
KSS Resources:	
Dedicated	SoS SEs, Specialist SoS SEs (performance, algorithms, interface, security…)
Shareable	Most
Sourced	PDE Teams
Managed resources	Specialty SoS SEs (performance, algorithms, interface, security…)
Activities:	

Description	*WIP Limit*	*Resource Type*	*Cohesion*
Capability Analysis		X-discipline team	Interruptible
Operational Concept Development		Internal, X-discipline team	Interruptible
Capability Requirements Creation		Internal, X-discipline team	Interruptible
Capability Requirement Development		Sourced	Interruptible
Special Engineering Services		Internal (managed)	Interruptible

Flow and Visibility:	
Additional CoS handled	Software Service CoS: One of the issues identified was the amount of time product tasks were blocked waiting for SoSE (CE) support. This CoS is applied to all Specialty Engineering Services work items from PTs with significant software components. The CoS is not interruptible and provides a guaranteed WIP capacity. Resource reallocation is allowed to meet this CoS.
Additional CoS introduced	None
Work Selection Value Adjustments	

Source-based	*CoS-based*	*Resource-based*	*Completion-based*
			Support to work associated with requirements or capabilities within 15% of completion are raised by 10% at selection cadence points

Goals	G1. Cost-effective and timely alternatives identified for new capabilities/capability enhancements G2. Adaptable, flexible, multi-purpose solutions provided for new capabilities/capability enhancements G3. Specialty engineering responses to software teams' SE requests do not create excessive delays in capability development G4. Provide quick response to changing needs and priorities
Questions answered	Q1. What work is currently blocked? Q2. What is the % of capabilities that are deployed within the desired timeframe? Q3. What is the predicted time to completion for "accepted" CE tasks (by class of service)? Q4. Where is capacity not meeting demand (by capability specialty engineering discipline)? Q5: Where is there excess capacity (by capability specialty engineering discipline)? Q6: What is the age of items in the CE backlog queues? Q7. What are the current CE WIP levels? Q8. What are the current CE backlog levels? Q9. What is the balance between CE WIP and CE backlog?
Data maintained/used	KSS1: Number/status of tasks in product-level queues (initial analysis, backlog, WIP, blocked) KSS2: Number of tasks in product-level queues that are blocking other tasks (e.g., dependent tasks) KSS3: Relationships between capabilities, requirements, and features at product level KSS4: Percentage of each in-process requirement already completed/deployed KSS5: Average User Support request task completion time
Information shared	Requirements allocation, status and deployment data; CE and PDE flow information

Accepting/Selecting Next Work Item. As requests come in for systems engineering services, whether front end work on new capabilities or work supporting other disciplines in their developing or sustaining activities, they are accepted, roughly estimated, possibly broken into smaller tasks, and valued. An additional CoS is assigned as necessary and then

the work items are added to the backlogs for the appropriate resource. Queue length limits are usually maintained for backlogs, and the level of the queue in terms of a percentage is a reasonable measure of demand. If the selection cadence is longer than daily, a WIP-limited "ready" queue can be added that allows the team to select a fixed number of tasks to accept and keep in the ready queue so work can begin immediately upon resource availability.

Allocating Resources and Team Development. Many CE tasks will require a team with expertise in one or more specialty engineering areas or may require collaborative support from one or more PDE Team SEs. The CE negotiates with the appropriate teams for the specific resources they need. CoS, nearness to completion of the requirement, and other factors are considered. For requests from software teams, the special software CoS is applied as described in the summary. Capability Requirements Development tasks are created, sourced to the various PDE Teams, and tracked to completion. Any negotiation required is accomplished before CE or the PDE Team accepts the work.

Completion and Disbursement. As CE completes ESM analysis work items, they are delivered directly to the ESM and identified as "done" on both the ESM and CE boards. Analysis tasks from PDTs are handled the same way. Work sourced to the PDE Teams may be completed and then deployed by the PDE Team. The PDE Team will share data regarding its status to update the CE KSS and Dashboard. There could be an activity to provide some form of requirement completion verification and validation within the CE KSS, but in this initial concept, this is handled within PDE. Data is also passed to the ESM dashboard.

KSS Review. Walking the CE KSS involves tracking the work in progress, identifying flow problems and blockages, resolving resource issues and blockages, and monitoring the demand queue so that when resources are available the next most valuable piece of work is accepted. The review tracks the WIP-level and demand for specialty resources to avoid blockage, overwork, or underutilization. Work items should be scanned for adjustment to work value or priority on completion-based criteria. Technical or PDE Team issues should be reviewed, and often it is good to include members of critical PDE Teams in the review.

The CE kanban board (Figure 5) is divided into two parts. The top shows the value stream for the activities that SE performs and the bottom tracks the specialty engineers.

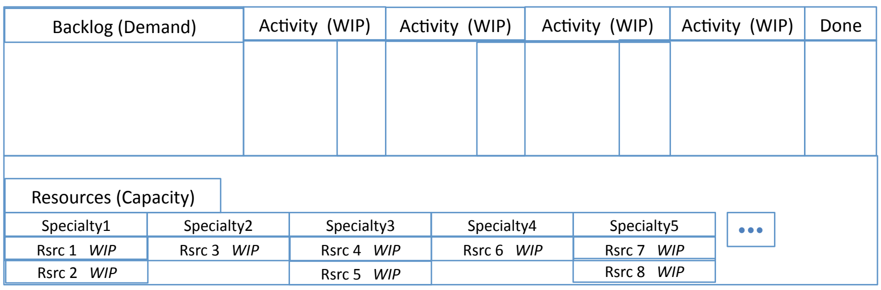

Figure 5. Notional CE Kanban Board

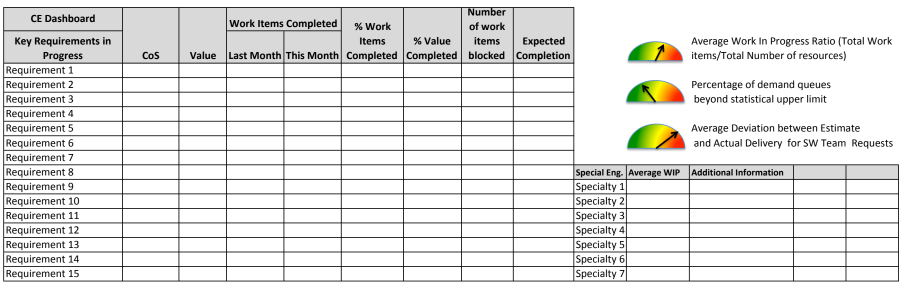

Figure 6. Notional CE Dashboard

User Support (US) KSS

User and site support personnel interact directly with the users and other operational stakeholders for the system of systems. They provide insight and triage for user requests; they aggregate and categorize desired capabilities or required maintenance actions, and forward them for resolution to the CE or PDE Teams as appropriate. The KSS is set up to manage the resources of the personnel handling the triage function and to identify critical issues rapidly. They track issues to completion and support information requests on the status of specific issues. This KSS is modeled on the system developed by Joshua Bloom at The Library Corporation, and the authors appreciate his support in this research.

Table 5. User Support KSS Template

User Support KSS			
Demand:			
Work sources	User requests		
Resources:			
Dedicated	Help Desk Personnel, SW/System Engineers		
Shareable	None		
Sourced	PDE Teams, CE		
Managed resource specialties	SW/System Engineers may be handled as managed resource specialists		
Activities:			
Description	*WIP Limit*	*Resource Type*	*Cohesion*
Call Reception and triage		Internal	Must complete
Secondary ticket review		Internal	Interruptible
Ticket assignment		Internal	Interruptible
Flow and Visibility:			
Additional CoS handled	None		
Additional CoS introduced	None		
Work Selection Value Adjustments			
Source-based	*CoS-based*	*Resource-based*	*Completion-based*
None	None	None	None
Goals	**Not yet addressed**		
Questions answered	**Not yet addressed**		
Data maintained	**Not yet addressed**		
Information shared	**Not yet addressed**		

Accepting/Selecting Next Work Item. US is the connection between the development system and the user population. Many user calls do not require development and are managed through the US KSS alone. Tickets for problems that require technical development work are written up and entered into the KSS demand queue. Initial estimations are of the "t-shirt size" variety and tickets are classified according to product, domain or other attribute. Any tickets critical to patient safety or require expedited activity are immediately handed off to the ESM, CE, and PDE teams to swarm and resolve quickly. Otherwise, initial classes of service are assigned.

Allocating Resources and Team Development. Once a ticket is entered into the demand queue, it is determined to be product specific and sent to a PDE team, it is determined to involve multiple products/domains and is entered into the ESM needs backlog as a systems of systems capability issue, or, it is not immediately understood and so sent to the SoS team to analyze and recommend action. All such tickets are maintained in the KSS as in-process and tracked through the system to completion so US can provide feedback on its status to users.

Completion and Disbursement. When PDE Team or CE development work is done, the US advises the ticket requestor(s) that the ticket has been resolved and provides a resolution to the user. This could be a software patch, workaround, or fix deployment date.

KSS Review is focused on the ability to effectively triage and assign tickets. Surveillance of the status of the technical work that entered through the US KSS provides a measure of response time to user requests and may be accompanied by user satisfaction information.

Because of the rapidity with which most help desk activities occur, our dashboard provides the information of a kanban board. The online dashboard from The Learning Corporation(Figure 7) is an example of how thee User Support Dashboard might look.

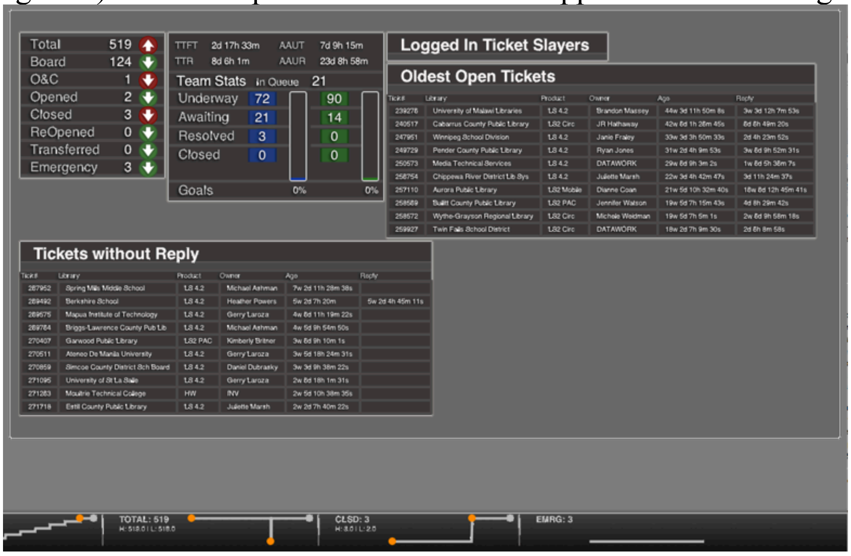

Figure 7. Help Desk Dashboard from The Learning Corporation

Product Team (PT) KSS

The PDE Product Teams are responsible for one or more of the Health Care System products. The teams include systems engineers, specialty engineers, software engineers, hardware engineers, and often subject matter experts that support feature determination and development. System of system capabilities may require multiple product teams to create or enhance features, implement similar features in different ways, or collaborate to develop a common solution for the specific systems. If CE is the heart of the system of systems, the product team is the arms and legs.

The PT KSS is focused on maintaining the product at a high level of effectiveness and evolving it to support system capabilities as well as product capabilities. There is always some tension among the new feature development, older feature enhancement, and typical maintenance that is required in a technology and safety critical environment. The KSS uses the various CoS defined for the system to manage flow so that major capability developments proceed at a reasonable pace without significant impact on ongoing project level work.

Accepting/Selecting Next Work Item. Selection at this level is all about balancing: the capacity with the demand, new work with ongoing activity, and SoS value with product value. While selection decisions are supported by the inherited value determination and CoSs, the product teams still negotiate the flow. The sourcing customers and PT members look at the mix of tasks in the demand queue, evaluating each according to the system values, product values and resources available, as well as considering what items represent the final parts of a requirement or capability. All teams will implement their selection strategy to match their own need for flow control.

Allocating Resources and Team Development. Most of the PT work is performed by groups of resources, often in a multi-discipline project team. Individual SE resources must also be available to participate in the cross-discipline/cross-system teams used in the CE in capability analysis, so there may be a reason to apply some sort of Project-level CoS that reserves some capacity for supporting those activities. Resource allocation and tracking strategies would differ from team to team depending on the availability of scarce resources and the mix and demand for specialty resources under their control.

Table 6. Product Team KSS Template

Product Team			
Demand:			
Work sources	US, CE, Internal, other PDE Teams		
Resources:			
Dedicated	SEs, HW and SW developers		
Shareable	SEs		
Sourced	SW Developers (SDPT), SoS Specialty Engineers (CE), Domain Specialists (DPT)		
Managed resource specialties	Varies by team		
Activities:			
Description	*WIP Limit*	*Resource Type*	*Cohesion*
Requirements analysis and feature definition		Internal, X-discipline team	Interruptible
Feature development and integration		Internal, Sourced	Interruptible
Requirements V&V		Internal, Sourced	Interruptible
Deployment		Internal, Sourced	Must complete
Flow and Visibility:			
Additional CoS handled	Software Service CoS: One of the issues identified was the amount of time product tasks were blocked waiting for SoSE (CE) support. This CoS is applied to all Specialty Engineering Services work items from PTs with significant software components. The CoS is not interruptible and provides a guaranteed WIP capacity. Resource reallocation is allowed to meet this CoS.		
Additional CoS introduced	Certification required – Applies where work is bundled to prevent costly recertification.		
Work Selection Value Adjustments			
Source-based	*CoS-based*	*Resource-based*	*Completion-based*
Varies by team	Varies by team	Varies by team	Support to work associated with requirements or capabilities within 15% of completion are raised by 10% at selection cadence points
Goals	G1. Capability-allocated requirements are developed and deployed according to value G2: Product requirements and features are allocated to increments and spins based on value G3. Product team responds quickly to changing product needs and priorities G4. Minimize workflow disruptions in product increments and spins G5. Minimize rework due to poorly understood capability requirements G6. Product team provides timely responses to user support issues/problems		
Questions answered	Q1. Value of product-level work currently blocked? Q2. What is the % of requirements completed within the desired timeframe? Q3. Where is PT capacity not meeting demand? Q4: Where is there excess PT capacity? Q5: How often is the average item age in product-level backlogs outside expected levels? Q6. What are the current product-level WIP levels? Q7. What are the current product-level backlog levels? Q8. What is the product-level response time to SW requests?		
Data maintained	KSS1: Flow data on Product Team* KSS2: Number/status of tasks in demand queues KSS3: Number of tasks in product-level activities that are blocking other tasks KSS4: Relationships between capabilities, requirements, and features at product level KSS5: Percentage of each in-process requirement already completed/deployed KSS6: Average User Support request task completion time **Includes CFD (throughput, WIP, Lead time), backlog level, resource utilization, blocked tasks, and similar data.*		
Information shared	Flow data on Product Team*		

Completion and Disbursement. Since PTs are responsible for integration, V&V and deployment, their kanban board addresses these activities. Data on status, acceptance and availability for inclusion of the various work items in completing capability implementation is always provided upstream to the sourcing KSS.

KSS Review. Walking the kanban board and reviewing the dashboard at the product level consists of looking for blocked work—resource conflict issues, sourcing delays, and rework are the main sources here. If the PT cannot complete work items within the statistical boundaries established over time, changes must be made quickly to balance demand.

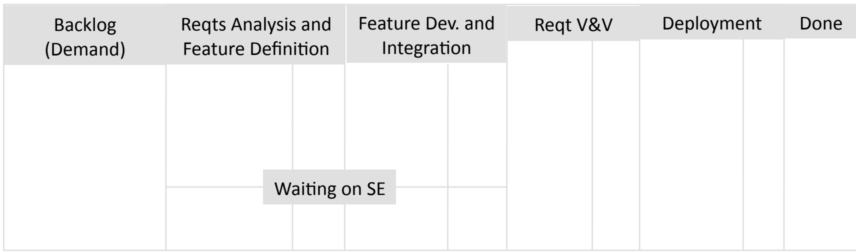

Figure 8. Notional PT Kanban Board

PT Dashboard			% Features	% Value	# Features	Expected
Requirements in Progress	CoS	Value	Completed	Completed	blocked	Completion
Requirement 1						
Requirement 2						
Requirement 3						
Requirement 4						
Features in Progress	Cos	Value	Reqt	Sourced to	Blocked?	
Feature 1						
Feature 2						
Feature 3						
Feature 4						
Feature 5						
Feature 6						
Feature 7						
Feature 8						
Feature 9						
Feature 10						
Feature 11						

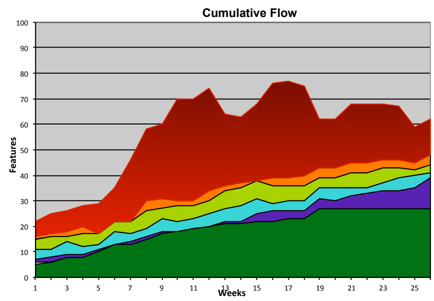

Figure 9. Notional PT Dashboard

Software Development Product Team (SPDT) KSS

While kanban in SW development is not new, the amount of SE activity and information provided at this level is significant in the Health Care scenario. Much of the performance reporting at the capability level is dependent on the WIP, WIP limit adjustments, lead times measured, statistical limits established, and process improvement activities in the SW development teams. Limited SoSE resources are one reason for the KSS Network.

Conclusions and Further Research

Much of this work has been engaged in thinking through all the various scenarios that exist in highly complex system development, sustainment and evolution. We are currently developing simulations of this KSS instantiation as well as others that have occurred to us throughout the research. We believe that KSS Networks can provide more realistic understanding of work in progress, organizational capacity and can bring some statistical probability to uncertain engineering activities. The irony is that KSS designs are uncertain as well. An experience that kanban users agree on is that pull systems are rarely "engineered" and usually evolve from the first instance in ways no one expected. For that reason, we are looking forward to sowing the seeds of our ideas into the systems engineering soil and seeing the unexpected but exciting harvest that grows out of them.

References

Anderson, David. (2010). Kanban: Successful Evolutionary Change for Your Technology Business. Sequim, WA: Blue Hole Press

Basili, Victor R. and Carolyn Seaman,(2010) "Metric-Based Quality Management", Software Engineering for Embedded Systems Series, Fraunhofer IESE, Kaiserslautern..

Boehm, Barry and Turner, Richard (2004). Balancing Agility and Discipline: A Guide for the Perplexed. Boston, MA: Addison Wesley.
J. Dahmann, G. Rebovich, J. Lane, R. Lowry, and K. Baldwin (2011); An Implementers' View of Systems Engineering for Systems of Systems, Proceedings of the IEEE International Systems Conference, 4-7 April, Montreal, Canada.
Larman C. and Vodde, B. (2009). Scaling Lean & Agile Development. Boston, MA: Addison Wesley.
Office of the Deputy Under Secretary of Defense for Acquisition and Technology, Systems and Software Engineering (2008). Systems Engineering Guide for Systems of Systems, Version 1.0. Washington, DC: ODUSD(A&T)SSE, 2008.
Poppendiek, Mary. (2007). Implementing Lean Software Development. Boston, MA: Addison Wesley.
Reinertsen, Donald G. (2010). The Principles of Product Development Flow. Redondo Beach, CA: Celeritas Publishing.
Turner, R., Madachy R., Ingold D., and Lane J. (2012) "Improving Systems Engineering Effectiveness in Rapid Response Development Environments," Proceedings of the International Conference on Software and System Process.
Turner, R., Madachy R., Ingold D., and Lane J., Anderson D., "Effectiveness of kanban approaches in systems engineering within rapid response environments," Proceedings of the Conference on Systems Engineering Research 2012, Procedia Computer Science, Vol. 8 - 2012, Elsevier, March, 2012.
Turner, Richard and Wade, J. (2011). "Lean Systems Engineering within System Design Activities," Proceedings of the 3rd Lean System and Software Conference, May 2-6, 2011, Los Angeles, CA.

Biography

Dr. Richard Turner has over thirty years of industry and government experience in systems, software and acquisition engineering. Currently a Distinguished Service Professor at the **Stevens Institute of Technology** in Hoboken, New Jersey, he is co-author of three books: *Balancing Agility and Discipline: A Guide for the Perplexed*, co-written with Barry Boehm, *CMMI Distilled*, and *CMMI Survival Guide: Just Enough Process Improvement*. Dr. Turner is also a Fellow of the Lean Systems Society.

Dr. Jo Ann Lane is Co-Director of the Center for Systems and Software Engineering as well as a Research Assistant Professor in the Daniel J. Epstein Department of Industrial and Systems Engineering at The University of Southern California. She was awarded the INCOSE Foundation/Stevens Doctoral Award for promising research in Systems Engineering and Integration. Prior to her current work, she was a key technical member of SAIC's Software and Systems Integration Group for over 20 years, responsible for the development and integration of software-intensive systems and systems of systems.

Dr. Raymond Madachy is an Associate Professor in the Systems Engineering Department at the Naval Postgraduate School. He has over 20 years of management and technical experience including Chief Science Officer at Cost Xpert Group and Manager of the Software Engineering Process Group at Litton Systems. He has over 90 publications including the book *Software Process Dynamics* and is a co-author of *Software Cost Estimation with COCOMO II*.

Dan Ingold is a Senior Research Analyst at the University of Southern California, Center for Systems and Software Engineering.